

# Rajalakshmi Engineering College

Name: Elango G  
Email: 241501055@rajalakshmi.edu.in  
Roll no: 241501055  
Phone: 7010568330  
Branch: REC  
Department: I AI & ML FA  
Batch: 2028  
Degree: B.E - AI & ML

Scan to verify results



## NeoColab\_REC\_CS23231\_DATA STRUCTURES

### REC\_DS using C\_Week 5\_COD\_Question 3

Attempt : 1  
Total Mark : 10  
Marks Obtained : 10

#### Section 1 : Coding

##### 1. Problem Statement

You are required to implement basic operations on a Binary Search Tree (BST), like insertion and searching.

Insertion: Given a list of integers, construct a Binary Search Tree by repeatedly inserting each integer into the tree according to the rules of a BST.

Searching: Given an integer, search for its presence in the constructed Binary Search Tree. Print whether the integer is found or not.

Write a program to calculate this efficiently.

##### ***Input Format***

The first line of input consists of an integer n, representing the number of nodes

in the binary search tree.

The second line consists of the values of the nodes, separated by space as integers.

The third line consists of an integer representing, the value that is to be searched.

### **Output Format**

The output prints, "Value <value> is found in the tree." if the given value is present, otherwise it prints: "Value <value> is not found in the tree."

Refer to the sample output for formatting specifications.

### **Sample Test Case**

Input: 7

8 3 10 1 6 14 23

6

Output: Value 6 is found in the tree.

### **Answer**

```
// You are using GCC
```

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
// Define the structure for the BST Node
```

```
struct Node {
```

```
    int data;
```

```
    struct Node* left;
```

```
    struct Node* right;
```

```
};
```

```
// Create a new node
```

```
struct Node* createNode(int value) {
```

```
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
```

```
    newNode->data = value;
```

```
    newNode->left = newNode->right = NULL;
```

```
    return newNode;
```

```
}
```

```
// Insert a value into the BST
struct Node* insert(struct Node* root, int value) {
    if (root == NULL) {
        return createNode(value);
    }

    if (value < root->data) {
        root->left = insert(root->left, value);
    } else {
        root->right = insert(root->right, value);
    }

    return root;
}
```

```
// Search for a value in the BST
int search(struct Node* root, int key) {
    if (root == NULL) {
        return 0; // Not found
    }

    if (root->data == key) {
        return 1; // Found
    } else if (key < root->data) {
        return search(root->left, key);
    } else {
        return search(root->right, key);
    }
}
```

```
int main() {
    int n, value, key;
    struct Node* root = NULL;

    scanf("%d", &n); // Number of nodes

    for (int i = 0; i < n; ++i) {
        scanf("%d", &value);
        root = insert(root, value);
    }

    scanf("%d", &key); // Value to search
```

```
if (search(root, key)) {  
    printf("Value %d is found in the tree.\n", key);  
} else {  
    printf("Value %d is not found in the tree.\n", key);  
}  
  
return 0;  
}
```

**Status :** Correct

**Marks :** 10/10